Implementasi Sistem Klasifikasi Fuzzy Berbasis Optimasi Koloni Semut untuk Diagnosa Penyakit Diabetes

Junian Triajianto, Yudhi Purwananto, dan Rully Soelaiman
Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111 Indonesia
e-mail: yudhi@if.its.ac.id

***Abstrak*—Diabetes merupakan penyakit metabolis yang ditandai dengan tingginya tingkat glukosa dalam darah. Banyak pasien yang tidak menyadari adanya gejala diabetes dalam dirinya. Oleh karena itu diperlukan sistem pakar yang bisa memberikan peringatan apakah seseorang menderita diabetes atau tidak. Dalam makalah ini diimplementasikan sistem klasifikasi fuzzy berbasis optimasi koloni semut untuk diagnosa penyakit diabetes. Sistem pakar ini menggunakan mesin inferensi fuzzy untuk melakukan prediksi penyakit diabetes. Aturan-aturan fuzzy yang digunakan untuk membentuk mesin inferensi fuzzy didapatkan dengan menerapkan optimasi koloni semut yang bertugas mempelajari data latih. Uji coba sistem dilakukan dengan menggunakan data set *Pima Indian Diabetes*. Performa terbaik yang dihasilkan oleh model adalah akurasi sebesar 78,55%, *precision* sebesar 79,61%, *recall* sebesar 78,56%, dan *F-measure* sebesar 79,02%.**

***Kata Kunci*—FCS-ANTMINER, Mesin Inferensi Fuzzy, Optimasi Koloni Semut, Sistem Pakar.**

I. PENDAHULUAN

Diagnosa penyakit diabetes, terutama pada penyakit diabetes mellitus, bukanlah hal yang mudah untuk dilakukan. Hal ini disebabkan karena beberapa gejala yang dialami pasien juga muncul pada penyakit lain. Jadi, selain untuk evaluasi hasil tes, dokter juga harus memperhatikan keputusan yang dibuat sebelumnya untuk pasien dengan kondisi yang sama dalam diagnosa penyakit diabetes. Dengan kata lain, dokter membutuhkan pengetahuan dan pengalaman untuk pengambilan keputusan yang tepat [1].

Data yang digunakan untuk diagnosa penyakit diabetes adalah data set *Pima Indian Diabetes* (PID) [2]. Sistem membangun model klasifikasi fuzzy berdasarkan 8 atribut dalam data set tersebut. Aturan-aturan fuzzy yang akan dijadikan bagian dalam model klasifikasi dibentuk dengan menerapkan algoritma optimasi koloni semut. Keseluruhan sistem klasifikasi ini disebut dengan FCS-ANTMINER [3].

Makalah ini bertujuan untuk melakukan prediksi diabetes berdasarkan data set PID dengan mengimplementasikan algoritma FCS-ANTMINER. Diharapkan dengan implementasi algoritma tersebut, didapatkan model klasifikasi dengan performa yang baik sehingga berguna dalam proses diagnosa penyakit diabetes.

Makalah ini terorganisasi menjadi 6 bab. Bab I menguraiakan latar belakang serta tujuan penulisan makalah. Bab II menguraikan dasar teori yang digunakan untuk implementasi sistem. Bab III menguraiakan perancangan perangkat lunak. Bab IV menguraiakan implementasi sistem. Hasil uji coba diuraiakan pada Bab V. Bab VI merupakan kesimpulan yang dapat diambil dari makalah ini.

II. DASAR TEORI

A. *Optimasi Koloni Semut*

Algoritma berbasis Optimasi Koloni Semut (ACO) pertama kali dikenalkan oleh Dorigo dan Stützle sebagai algoritma heuristik untuk menyelesaikan permasalahan kombinatorial dengan pencarian lokal yang efisien. ACO adalah prosedur pencarian stokastik yang terinspirasi dari tingkah laku koloni semut di alam dalam mencari jalur terpendek dari sumber makanan menuju sarang mereka. Mereka meninggalkan jejak di atas tanah berupa substansi kimia bernama feromon saat berjalan. Feromon berguna dalam komunikasi antar semut. Setiap semut bisa mencium feromon, dan ketika mereka memilih jalur, mereka cenderung lebih memilih jalur dengan kadar feromon yang tinggi. Jika tidak ada feromon yang tersisa di atas tanah, maka semut cenderung memilih jalur secara acak. Feromon mengalami penguapan setiap waktu. Oleh karena itu, kadar feromon pada jalur yang lebih pendek akan menguap lebih lama daripada jalur yang lebih panjang karena pada jalur yang lebih pendek lebih sering dilewati semut. Dari kadar feromon yang terkandung di atas jalan, bisa diambil jalur terpendek yang paling mendekati solusi global optimal [4].

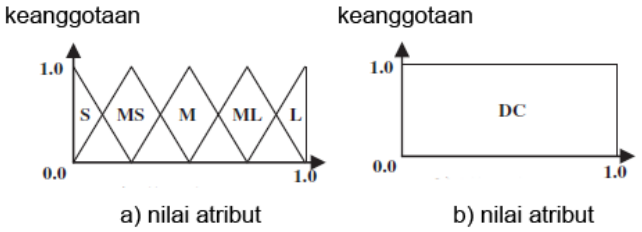
Perlu diketahui bahwa semut tiruan ini memiliki beberapa kelebihan dibandingkan semut asli, di antaranya adalah:

a. Semut tiruan memiliki ingatan.
b. Semut tiruan tidak sepenuhnya buta.
c. Lingkungan semut tiruan bersifat diskrit.

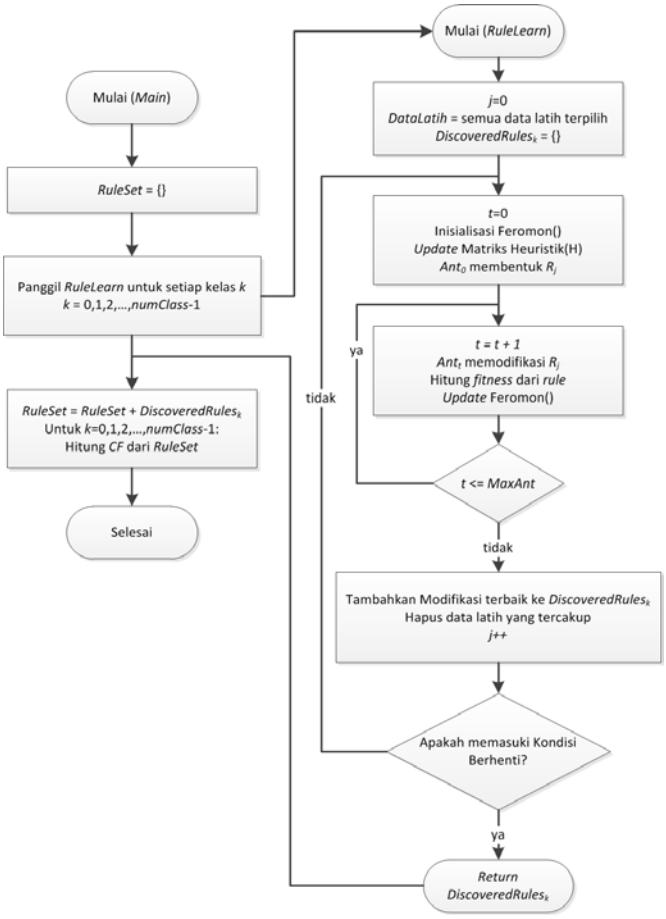
Selain digunakan untuk mencari jalur terpendek, algoritma

Aturan R_j jika x_1 adalah A_{j1} dan … dan x_n adalah A_{jn}
maka Kelas adalah C_j
dengan CF = CF_j

Gambar 1. Bentuk aturan jika-maka fuzzy.



Gambar 2. Fungsi keanggotaan fuzzy yang digunakan. (a) 1: *small*, 2: *medium small*, 3: *medium*, 4: *medium large*, 5: *large*, (b) 0: *don't care*.



Gambar 3. Diagram alur FCS-ANTMINER.

ACO juga bisa digunakan dalam bidang penggalian data untuk melakukan ekstraksi aturan-aturan klasifikasi. Parpinelli, Lopes, dan Freitas adalah orang-orang yang pertama kali menggunakan algoritma ACO untuk menyelesaikan permasalahan klasifikasi. Algoritma ini biasa disebut dengan ANTMINER [5].

B. Sistem Klasifikasi Fuzzy Berbasis Optimasi Koloni Semut

Sistem klasifikasi fuzzy berbasis optimasi koloni semut (FCS-ANTMINER) merupakan suatu metode untuk melakukan ekstraksi aturan-aturan fuzzy dari data latih yang dipelajari dengan menggunakan algoritma ACO. Metode ini hanya bisa melakukan pembelajaran untuk data set yang memiliki atribut bertipe numerik [3].

FCS-ANTMINER memiliki dua tahap utama, yaitu tahap pembelajaran dan tahap uji. Pada tahap pembelajaran, algoritma ACO dieksekusi untuk membentuk kumpulan aturan fuzzy yang ditunjukkan pada Gambar 1.

R_j merupakan label dari aturan jika-maka fuzzy ke-j, x_1, …, x_n merupakan fitur dalam aturan fuzzy, A_{j1}, …, A_{jn} merupakan nilai dari fitur dalam aturan fuzzy yang harus dipenuhi dalam rentang [0, 1], C_j merupakan label kelas dari aturan fuzzy yang bersangkutan, dan CF_j merupakan tingkat keyakinan dari aturan jika-maka fuzzy R_j.

Setiap elemen dalam aturan fuzzy direpresentasikan dalam bentuk nilai linguistik seperti pada Gambar 2. Fungsi keanggotaan fuzzy yang digunakan dalam FCS-ANTMINER merupakan partisi kurva segitiga sesuai Gambar 2.

FCS-ANTMINER melakukan pembelajaran aturan yang terkait pada setiap kelas secara terpisah. Oleh karena itu, jika ada kelas sejumlah c, maka akan terbentuk kumpulan aturan fuzzy sejumlah c yang dipelajari secara independen. Kombinasi dari seluruh kumpulan aturan fuzzy ini akan menjadi elemen pembentuk sistem klasifikasi fuzzy akhir. Setelah model klasifikasi terbentuk, langkah berikutnya adalah melakukan penghitungan performa dengan menggunakan mesin inferensi fuzzy. Alur pembelajaran ini digambarkan dengan diagram alur pada Gambar 3.

Pembentukan Aturan Fuzzy

Pada versi ANTMINER sebelumnya, setiap semut membuat aturan dan mencoba untuk memilih aturan terbaik sehingga setiap semut akan saling bersaing [5]. Kompetisi antar semut akan menyebabkan beberapa semut memilih aturan berakurasi tinggi, sedangkan semut lainnya akan kesulitan memilih jalur dengan kualitas aturan yang tinggi. Hal ini menyebabkan sistem klasifikasi akhir memiliki aturan dengan kualitas yang tinggi dan rendah, sehingga sistem kemungkinan akan gagal dalam melakukan klasifikasi data dalam jumlah yang signifikan. Penyebab utama hal ini adalah kurangnya kooperasi dalam koloni semut. Oleh sebab itu, kooperasi dan kompetisi dalam koloni semut diseimbangkan dalam FCS-ANTMINER.

Proses pembelajaran aturan fuzzy dilakukan secara terpisah untuk setiap kelas. Dengan kata lain, pemanggilan fungsi *RuleLearn* dilakukan sebanyak jumlah kelas yang ada. Mula-mula *DiscoveredRules*, variabel untuk menyimpan aturan yang dibentuk oleh koloni semut, dikosongkan. Tabel Feromon diinisialisasi dengan Persamaan (1).

$$\tau_{i,j}(t=0) = \frac{1}{\sum_{i=1}^{a} b_i} \quad (1)$$

a merupakan jumlah atribut

b_i merupakan jumlah *antecendent* fuzzy set dari attribut *i*

Ant_0 (semut pertama) mula-mula membentuk aturan R_j secara acak dengan menambahkan satu *term* (*term* terdiri atas atribut, operator, dan nilai) dalam satu waktu, serta nilai elemen dari matriks heuristik diperhitungkan. Nilai heuristik ini akan digunakan semut sebagai informasi heuristik.

Dalam perulangan fungsi *RuleLearn*, ant_t (t=1,…,*MaxAnt*-1) memodifikasi aturan R_j berdasarkan parameter *MaxChange* yang artinya ant_t hanya boleh memodifikasi aturan R_j hingga *term* yang ditambahkan maksimal sebanyak *MaxChange*. Setelah semut memodifikasi aturan R_j maka langkah berikutnya adalah menghitung kualitas R_j dan melakukan pembaruan feromon berdasarkan kualitas R_j. Setelah semua semut selesai memodifikasi R_j, aturan fuzzy dari semut dengan modifikasi terbaik ditambahkan ke dalam *DiscoveredRules*. Selanjutnya semua sampel yang tercakup dalam aturan terbaik tadi (sampel yang memenuhi *antecedent* aturan dan memiliki label kelas sesuai dengan kelas yang sedang dilatih) dihapus dari data latih.

Langkah di atas terus diulangi untuk mempelajari aturan yang tersisa dari kelas yang sedang dilatih. Jika memenuhi kondisi berhenti, fungsi *RuleLearn* mengembalikan aturan-aturan fuzzy yang telah dipelajari ke fungsi utama. Pada akhirnya semua aturan yang dipelajari dari masing-masing kelas akan digunakan sebagai model deteksi untuk diagnosa penyakit diabetes.

Modifikasi Aturan Fuzzy

Pada iterasi pertama ($t = 0$), ant_0 membentuk aturan R_h dan pada iterasi berikutnya ($t \geq 1$) semut-semut memodifikasi aturan R_h. Jumlah *term* maksimal yang bisa dimodifikasi oleh semut di setiap iterasi ($t \geq 1$) ditentukan oleh parameter *MaxChange*. Parameter ini bisa bernilai berapapun selama tidak melebihi jumlah atribut ($0 < MaxChange \leq$ jumlah atribut). Misalkan $term_{i,j}$ adalah kondisi aturan dari bentuk (T_i adalah A_j) dengan T_i adalah atribut ke-*i* dan A_j adalah *antecedent* ke-*j* (misalkan *Age* adalah *S*) himpunan fuzzy. Setiap semut memilih $term_{i,j}$ berdasarkan tabel probabilitas yang nilainya dihitung dari Persamaan (2).

$$P_{i,j} = \frac{\tau_{i,j}(t).\eta_{i,j}}{\sum_{i}^{a}\sum_{j}^{b_i}\tau_{i,j}(t).\eta_{i,j}, \forall i \in I} \quad (2)$$

a merupakan jumlah atribut.

b_i adalah jumlah *antecendent* himpunan fuzzy dari atribut ke-*i*.

I adalah kumpulan atribut yang belum dipilih oleh semut.

$\eta_{i,j}$ merupakan nilai heuristik untuk $term_{i,j}$.

Pada Persamaan (2), $\tau_{i,j}(t)$ adalah jumlah feromon saat waktu *t* pada jalur antara atribut ke-*i* dan *antecedent* himpunan fuzzy ke-*j*. Semut akan cenderung memilih jalur dengan kadar feromon yang tinggi secara acak.

Feromon yang ditinggalkan semut nilainya tergantung pada kualitas aturan yang dimodifikasinya. Oleh karena itu, jika aturan R_h yang telah dimodifikasi kualitasnya meningkat, maka jalur ini akan diikuti lebih banyak semut. Dengan begitu, maka jumlah feromon pada $term_{i,j}$ akan semakin meningkat. Implikasinya, nilai probabilitas untuk dipilih juga semakin tinggi. Perlu dicatat bahwa setiap semut memiliki ingatan dan hanya bisa memodifikasi setiap *term* pada R_h sekali.

Informasi Heuristik

Setiap semut memodifikasi setiap *term* dari aturan berdasarkan informasi heuristik dan jumlah feromon. FCS-ANTMINER menggunakan matriks dua dimensi, dengan nama $H = \{H_1, H_2, \ldots, H_{numClass}\}$, untuk menyimpan informasi heuristik. Untuk setiap kelas *k* disediakan matriks H_k dengan baris sebagai atribut dan kolom sebagai nilai fuzzy. Setiap kali aturan baru ditambahkan ke dalam *DiscoveredRules* dan sampel yang tercakup dihapus dari data latih, nilai matriks *H* diperbarui dan tidak diubah selama proses modifikasi aturan. Matriks ini membantu semut memilih *term* yang relevan dan membuat aturan berkualitas tinggi.

Setiap elemen pada H_k, yaitu $h_{i,j}$ ($j > 1$) berisi jumlah sampel yang belum ditemukan polanya dengan label kelas *k* dan nilai untuk atribut *i* sama dengan nilai fuzzy ke-*j*, *j*=1,2,…,6 (DC adalah nilai fuzzy pertama, S adalah nilai fuzzy kedua, dan seterusnya).

Kolom pertama pada matriks ($h_{i,1}$ *i*=1,2,…,*n*) merupakan probabilitas *don't care* (DC) dari fitur. Maksud dari DC adalah bahwa ada kemungkinan atribut ke-*i* tidak dihiraukan dalam aturan fuzzy. Nilai DC setiap atribut dihitung dengan entropi berdasarkan Persamaan (3) dan (4).

$$E_{i,1} = -\sum_{j=2}^{6} P(h_{i,j} \mid sumh_i).\log P(h_{i,j} \mid sumh_i) \quad (3)$$
$$i = 1,2,...,n.$$

$$h_{i,1} = 1 - E_{i,1} \quad (4)$$

$sumh_i$ merupakan jumlah semua nilai non-DC pada atribut$_i$

$P(h_{i,j} \mid sumh_i)$ merupakan probabilitas empiris observasi $h_{i,j}$

Langkah penting berikutnya adalah dilakukan normalisasi untuk semua kolom pada matriks dengan menggunakan Persamaan (5).

$$\eta_{i,j} = \frac{h_{i,j}}{Max(h_{i,j}), quad \forall i = 1,2,...,n} \quad (5)$$

Fungsi Komputasi Kualitas

Ketika semut selesai memodifikasi aturan R_h, dilakukan perhitungan fungsi kompurasi kualitas. Kualitas dari aturan yang dimodifikasi dihitung dengan Persamaan (6).

$$Q_j = \frac{TP}{TP + FN} . \frac{TN}{TN + FP} \quad (6)$$

TP, *True Positives*, merupakan jumlah kasus di mana data yang masuk dalam cakupan prediksi dengan nilai prediksi sama dengan nilai aktual. FP, *False Positives*, merupakan jumlah kasus di mana data yang masuk dalam cakupan prediksi dengan nilai prediksi tidak sama dengan nilai aktual. FN, *False Negatives*, merupakan jumlah kasus di mana data yang tidak masuk dalam cakupan prediksi dengan nilai prediksi sama dengan nilai aktual. TN, *True Negatives*, merupakan jumlah kasus di mana data yang tidak masuk dalam cakupan prediksi dengan nilai prediksi tidak sama dengan nilai aktual.

Aturan Pembaruan Feromon

Setiap kali semut selesai melakukan modifikasi aturan R_h, maka selanjutnya dilakukan perhitungan kualitas aturan. Jika kualitas aturan hasil modifikasi semut meningkat, dilakukan pembaruan tabel feromon dengan Persamaan (7) dan (8).

$$\Delta_Q = Q_i^{SetelahModifikasi} - Q_i^{SebelumModifikasi} \quad (7)$$

$$\tau_{i,j}(t+1) = \tau_{i,j}(t) + \tau_{i,j}(t)(\Delta_i^Q . C) \quad (8)$$

Δ_Q merupakan selisih kualitas setelah modifikasi dengan kualitas sebelum modifikasi. *C* merupakan parameter untuk tingkatan pengaruh pembaruan feromon (dalam kasus ini digunakan nilai 0.5)

Feromon untuk *term* yang tidak dimodifikasi semut nilainya dikurangi dengan cara membagi setiap nilai feromon $\tau_{i,j}$ dengan jumlah keseluruhan feromon $\tau_{i,j}$ [5].

Kondisi Berhenti

Kondisi berhenti merupakan suatu kondisi di mana perulangan pada fungsi *RuleLearn* berakhir sesuai dengan definisi pengguna. Contoh kondisi berhenti adalah digunakan jumlah iterasi yang tetap atau jumlah sampel minimal yang belum tercakup. Kondisi berhenti FCS-ANTMINER adalah ketika jumlah sampel yang belum tercakup bernilai tetap dalam dua iterasi berurutan atau ketika hampir semua semut memilih jalur yang sama.

C. Mesin Inferensi Fuzzy

Diasumsikan permasalahan klasifikasi terdiri dari kelas sejumlah *c* dalam data set berdimensi *n* dengan keseluruhan atribut bertipe kontinu. Diasumsikan juga vektor sejumlah *M* $x_p = (x_{p1}, x_{p2}, .. x_{pn})$ sebagai data latih dengan kelas sejumlah *c* ($c << M$).

Ketika FCS-ANTMINER membentuk sekumpulan aturan fuzzy dengan menggunakan data latih sejumlah *M*, untuk melakukan klasifikasi data uji, mula-mula dilakukan perhitungan tingkat keyakinan (CF) untuk masing-masing aturan jika-maka fuzzy.

Langkah pertama adalah dilakukan perhitungan tingkat kesesuaian masing-masing data latih $x_p = (x_{p1}, x_{p2}, .. x_{pn})$ terhadap aturan jika-maka fuzzy R_j dengan Persamaan (9).

$$\mu_j(x_p) = \mu_{j1}(x_{p1}) \times ... \times \mu_{jn}(x_{pm}) \quad p = 1,2,3,...,m \quad (9)$$

$\mu_{ji}(x_{pi})$ merupakan fungsi keanggotaan pada atribut ke-*i* pada data latih ke-*p* dan *m* merupakan jumlah sampel dalam data latih.

Langkah kedua adalah dilakukan perhitungan jumlah relatif terhadap tingkat kesesuaian yang telah dihitung pada Langkah pertama dengan menggunakan Persamaan (10) untuk setiap kelas.

$$\beta_{classh}(R_j) = \sum_{x_p \in classh} \frac{\mu_j(x_p)}{N_{classh}}, h = 1,2,...,c \quad (10)$$

Langkah ketiga adalah melakukan perhitungan tingkat keyakinan dari aturan jika-maka fuzzy dengan menggunakan Persamaan (11).

$$CF_j = \frac{\beta_{classh_j}(R_j) - \overline{\beta}}{\sum_{h=1}^{c} \beta_{classh}(R_j)} \quad (11)$$

$$\overline{\beta} = \frac{\sum_{h \neq \overline{h_j}} \beta_{classh}(R_j)}{(c-1)} \quad (12)$$

Setelah semua tingkat keyakinan dari masing-masing aturan jika-maka fuzzy dihitung, bisa dilakukan uji coba model dengan data uji. Tugas utama dari sistem klasifikasi FCS-ANTMINER adalah untuk membentuk kombinasi *antecedent* himpunan fuzzy untuk membentuk himpunan aturan *S* dengan kemampuan klasifikasi yang tinggi. Jika ada pola data uji $x_p = (x_{p1}, x_{p2}, \ldots, x_{pn})$, maka untuk mengklasifikasi pola data uji ini dipilih satu aturan terbaik R_j dari *S*. yang ditentukan dengan Persamaan (13). Aturan yang menang adalah aturan dengan nilai perkalian tingkat

kesesuaian dan tingkat keyakinan CF_j maksimal [6].

$$\mu_j(x_p).CF_j = \max\{\mu_j(x_p).CF_j \mid R_j\} \quad (13)$$

III. PERANCANGAN PERANGKAT LUNAK

Rancangan secara umum perangkat lunak prediksi diabetes dengan FCS-ANTMINER dimulai dengan masukan data set sebagai data latih ke dalam sistem. Kemudian nilai-nilai parameter yang digunakan dalam FCS-ANTMINER ditentukan oleh pengguna sebelum memasuki tahap pembelajaran (misalkan parameter *MaxChange*, *MaxAnt*, dan sebagainya).

Sebelum melakukan pembelajaran, data set yang digunakan sistem harus dinormalisasi terlebih dahulu dengan (14). Lalu dibuat himpunan fuzzy dengan fungsi keanggotaan seperti pada Gambar 2. Setelah itu dilakukan tahap pembelajaran dengan algortima FCS-ANTMINER.

$$Normalize(X) = \frac{X - X\min}{X\max - X\min} \quad (14)$$

X merupakan data yang dinormalisasi, X_{min} adalah nilai terkecil dari kolom, dan X_{max} adalah nilai terbesar dari kolom.

Setelah model prediksi terbentuk, langkah berikutnya adalah menguji performa dari model dengan menggunakan *k-fold cross validation*. Jika performa yang dihasilkan masih belum memuaskan, langkah pembelajaran bisa dilakukan lagi dengan mengubah parameter sebelumnya. Jika performa yang dihasilkan sudah bagus maka bisa dilakukan prediksi diabetes terhadap data masukan yang belum terklasifikasi.

IV. IMPLEMENTASI

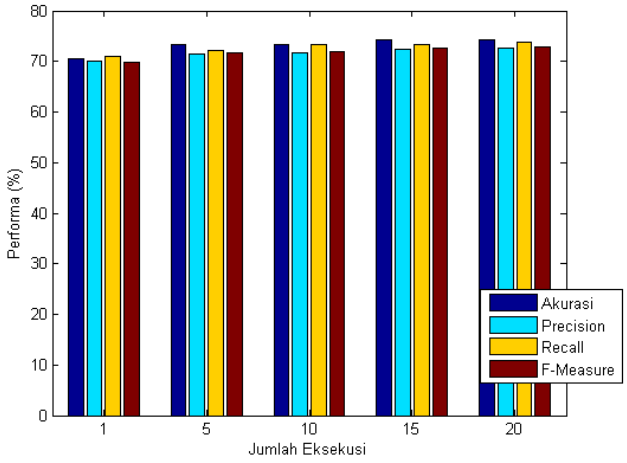
Perangkat lunak sistem pakar ini dibangun dengan menggunakan perangkat pengembang Microsoft Visual Sudio 2010. Bahasa pemrograman yang digunakan adalah C#. Antarmuka dari aplikasi dibentuk dengan menggunakan WPF.

Sebelum dilakukan proses latih dan uji, perlu disiapkan berkas yang berisi data latih. Data latih yang digunakan merupakan data latih PID. Data latih tersebut sebelumnya harus dinormalisasi terlebih dahulu sebelum digunakan.

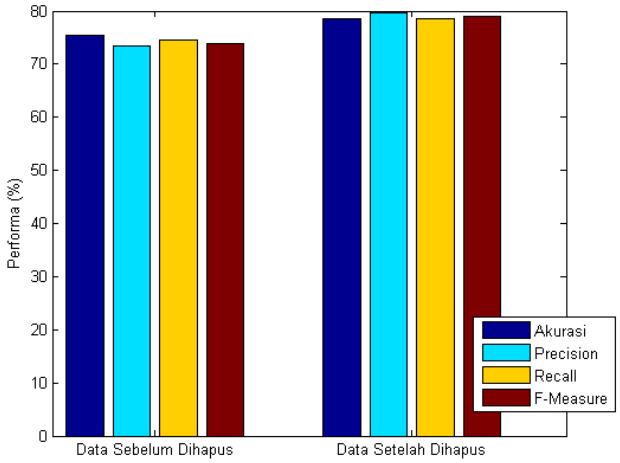
Setelah itu dibuat fungsi keanggotaan fuzzy berdasarkan Gambar 1. Fungsi keanggotaan fuzzy ini nantinya digunakan dalam pembelajaran FCS-ANTMINER.

Langkah berikutnya adalah melakukan pembelajaran dengan data set yang disediakan. Pembelajaran dilakukan sesuai dengan Gambar 3.

Setelah model terbentuk, langkah berikutnya adalah melakukan uji coba sistem. Untuk setiap uji coba dilakukan pengukuran dan perbandingan performa.



Gambar 4. Grafik performa model berdasarkan eksekusi ACO.

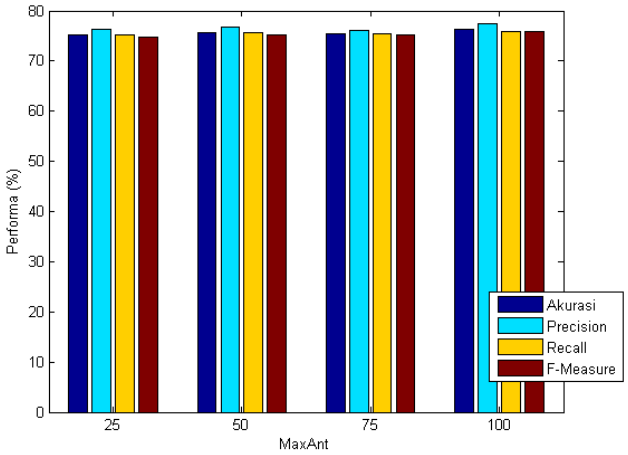


Gambar 5. Grafik performa model antara data set yang tidak seimbang dengan data set yang seimbang.

V. UJI COBA

Pada bagian ini dijelaskan skenario uji coba yang telah dilakukan. Skenario uji coba dibagi menjadi tiga skenario. Skenario pertama adalah mengganti jumlah eksekusi algoritma ACO untuk mendapatkan kumpulan aturan fuzzy, skenario kedua adalah melakukan menggunakan data set dengan distribusi tiap kelas yang seimbang, dan skenario ketiga adalah mengganti parameter *MaxAnt*. Dari setiap skenario dilakukan perbandingan untuk mengerahui pengaruh parameter pada performa model yang dihasilkan.

Nilai performa masing-masing uji coba diukur dalam empat parameter, yaitu akurasi, *precision*, *recall*, dan *F-measure*. Masing-masing nilai performa secara berurutan dihitung dengan Persamaan (15) hingga (18).



Gambar 6. Grafik performa model berdasarkan jumlah *MaxAnt*.

$$Akurasi = \frac{TP + TN}{TP + TN + FN + FP} \quad (15)$$

$$precision = \frac{TP}{TP + FP} \quad (16)$$

$$recall = \frac{TP}{TP + FN} \quad (17)$$

$$FMeasure = \frac{2 \times precision \times recall}{precision + recall} \quad (18)$$

Gambar 4 menunjukkan bahwa semakin banyak eksekusi ACO yang dilakukan, maka akan semakin baik performa model yang didapat. Gambar 5 menunjukkan bahwa jika data set yang digunakan seimbang, maka akan menghasilkan performa yang lebih baik daripada data yang tidak seimbang. Gambar 6 menunjukkan bahwa semakin banyak jumlah semut yang digunakan, maka performa yang diperoleh juga meningkat. Namun kenaikan ini tidak signifikan sehingga pemilihan nilai *MaxAnt* disarankan tidak melebihi 1000.

VI. KESIMPULAN

Berdasarkan hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil, yaitu:

1) Model yang dihasilkan FCS-ANTMINER memiliki performa dengan akurasi sebesar 78,55% , *precision* sebesar 79,61%, *recall* sebesar 78,56%, dan *F-measure* sebesar 79,02%.
2) Model FCS-ANTMINER akan memberikan performa yang maksimal ketika dilakukan algoritma ACO antara 15 hingga 50 dengan data set yang memiliki distribusi kelas yang seimbang.

DAFTAR PUSTAKA

[1] "American Diabetes Association," 2010. [Online]. Available: http://www.diabetes.org/diabetes-basics/. [Diakses Agustus 2012].
[2] A. Frank dan A. Asuncion, University of California, Irvine, School of Information and Computer Sciences, 2010. [Online]. Available: http://archive.ics.uci.edu/ml.
[3] M. F. Ganji dan M. S. Abadeh, "A fuzzy classification system based on Ant Colony Optimization for diabetes disease diagnosis," *Expert Systems with Applications*, (2011) 14650-14659.
[4] M. Dorigo, M. Birattari dan T. Stutzle, "Ant colony optimization," *Computational Intelligence Magazine, IEEE*, Vol. 1, No. 4, (2006) 28-39.
[5] R. Parpinelli, H. Lopes dan A. Freitas, "Data mining with an ant colony optimization algorithm," *Evolutionary Computation, IEEE Transactions on*, Vol. 6, No. 4, (2002) 321-332.
[6] K. Nozaki, H. Ishibuchi dan H. Tanaka, "Adaptive fuzzy rule-based classification systems," *Fuzzy Systems, IEEE Transactions on*, vol. 4, no. 3, (1996) 238-250.